Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

Claim 1 (currently amended): A method comprising:

dynamically obtaining one or more program operators from source code in a pre-compiler of a compiler module; and

applying data transformation to a portion of the source code in the pre-compiler based on one of said one or more program operators to provide encrypting compiler-generated code.

Claim 2 (original): The method of claim 1, including mixing the encrypting compiler-generated code with the source code other than said portion before compilation.

Claim 3 (currently amended): The method of claim 1, further comprising dynamically deriving from the source code at least one compiler-generated operator for said data transformation.

Claim 4 (original): The method of claim 3, further comprising performing encryption using at least one of said at least one compiler-generated operator and said at least one of said one or more program operators.

Claim 5 (original): The method of claim 1, further comprising selectively encrypting one or more regions of the source code with a custom cipher formed from said at least one of said one or more program operators.

Claim 6 (original): The method of claim 1, including:

determining at least two references for each variable of a variable pair to selectively encrypt and decrypt data in between said at least two references; and

associating at least two data values with each variable of the variable pair for encryption of the data in a first transformation and decryption of the data in a second transformation.

Claim 7 (original): The method of claim 6, further comprising:

iteratively forming matching pairs of said data values for each variable of the variable pair; and

creating interlocking Feistal networks in each iteration involving a different matching pair of said data values.

Claim 8 (original): The method of claim 7, further comprising:

enabling detection of usage of one or more redundant computations in the interlocking Feistal networks; and

in response to a change in at least one of the one or more redundant computations, provisioning for corruption of unrelated data values relative to said data values.

Claim 9 (original): A method comprising:

analyzing flow of data in source code having one or more program operators to determine matching references to a pair of variables;

determining a block of the source code in which said pair of variables is not used; associating the matching references based on a heuristic to provide data encryption to modify a portion of the source code into encrypting compiler-generated code; and

mixing the encrypting compiler-generated code with the source code.

Claim 10 (original): The method of claim 9, wherein analyzing flow of data further including:

detecting a first region of the source code in which use of a stored value for at least one variable of said pair of variables occurs; and

detecting a second region of the source code in which the stored value is defined for the at least one variable of said pair of variables.

Claim 11 (original): The method of claim 9, including utilizing the heuristic to enhance obfuscation of the encrypting compiler-generated code within the source code using at least one of said one or more program operators.

Claim 12 (original): A method comprising:

identifying a first reference point and a second reference point within a set of blocks of source code having one or more program operators;

associating an encryption code in proximity to the first reference point and associating a decryption code in proximity to the second reference point; and

compiling a portion of the source code into encrypting compiler-generated code to mix with the source code other than said portion.

Claim 13 (original): The method of claim 12, further comprising:

customizing a cipher based on at least one of said one or more program operators;

selecting a block from the set of blocks, the block containing a first variable having a maximum distance over the set of blocks, and a second variable having a next maximal distance in the same block;

providing the encryption code to encrypt data in between a pair of references to the first and second variables; and

providing the decryption code to decrypt said data.

Claim 14 (original): The method of claim 13, further comprising recompiling the encrypting compiler-generated code with the source code other than said portion into tamper resistant object code.

Claim 15 (original): The method of claim 13, including:

deriving from the source code at least one compiler-generated operator for data flow transformation; and

using at least one of said at least one compiler-generated operator and said at least one of said one or more program operators to provide the encryption code.

Claim 16 (original): An article comprising a medium storing instructions that, if executed enable a system to:

dynamically obtain one or more program operators from source code; and

apply data transformation to a portion of the source code based on one of said one or more program operators to form encrypting compiler-generated code.

Claim 17 (original): The article of claim 16, further comprising instructions that if executed enable the system to mix the encrypting compiler-generated code with the source code other than said portion.

Claim 18 (original): The article of claim 16, further comprising instructions that, if executed enable the system to use at least one compiler-generated operator and said at least one of said one or more program operators for encryption.

Claim 19 (original): The article of claim 16, further comprising instructions that, if executed enable the system to selectively encrypt one or more regions of the source code with a cipher formed from said at least one of said one or more program operators.

Claim 20 (original): An apparatus comprising:

an analyzer to perform data flow analysis of source code to dynamically obtain one or more program operators therefrom; and

a code transformer coupled to said analyzer to apply data transformation to select a selected region of the source code in which to provide encrypting compiler-generated code based on one of said one or more program operators.

Claim 21 (original): The apparatus of claim 20, further comprising a cipher based on said at least one of one or more program operators.

Claim 22 (original): The apparatus of claim 20, further comprising an encryption engine to selectively encrypt and decrypt the selected region based on references to a variable identified in the selected region.

Claim 23 (original): The apparatus of claim 22, further comprising a heuristic to select the selected region and the references.

Claim 24 (original): A system comprising:

a dynamic random access memory having source code stored therein;

an analyzer to perform data flow analysis of the source code to dynamically obtain one or more program operators therefrom; and

a code transformer coupled to said analyzer to apply data transformation to select a selected region of the source code to provide encrypting compiler-generated code based on one of said one or more program operators.

Claim 25 (original): The system of claim 24, further comprising a cipher based on said at least one of one or more program operators.

Claim 26 (original): The system of claim 24, further comprising an encryption engine to selectively encrypt and decrypt the selected region based on references to a variable identified in the selected region.

Claim 27 (original): The system of claim 26, further comprising a heuristic to select the selected region and the references.